# CSCE 785, Fall 2024
# Homework 4, due November 15, 2024 before 11:30pm

## 1 Written Exercises

Do Exercises 3.16, 3.18, 4.5, 4.8 (optional), 4.9, 4.1, 4.12 (optional), 6.2, 6.4 (optional)

## 2 IonQ Platform Exercises

Implement the quantum Fourier transform on two and three qubits. For the three-qubit case, use the decomposition given in the notes and in class where $m = 2$ (and $n = 3$).

Jypyter notebooks are preferred, but straight text-only Qiskit code is also acceptable. One way or the other, I'll need to see a graphical representation of the circuit, which you may either draw by hand or have Qiskit do it for you in a jupyter notebook. You may either leave the swaps in place or move them all to the end of the circuit, which will alter the positions of some of the other gates.

Just for the 2-qubit QFT, test it on one of IonQ's QPUs (after debugging on a simulator). Run the circuit with each of the four computational basis states[1] and for each input, measure the output qubits in both the computational basis and the Hadamard basis. (Measuring in the Hadamard basis is equivalent to applying an $H$ gate before measuring in the computational basis.) Keep the number of shots per run below 500. Try, say, 200 shots for each run.

You do not need to test the 3-qubit QFT.

[1] All qubits start in the $|0\rangle$ state, so to get the $|1\rangle$ state, add an initial $X$ gate to the qubit.

## 2.1 A Graphical Circuit Builder

This is only a suggestion and is not required.

IBM Quantum has an online tool that lets you build a quantum circuit by dragging gates from a palette onto wires, automatically creating the corresponding Qiskit code—which is also directly editable—as you go. You may need to create an account with them (go to `https://quantum.ibm.com`), but I believe that doing so is free and easy; you just won't get priority access to their QPUs. Once you've done this, look for a link on the homepage to "IBM Quantum Composer." You can export the code to a file and use it (with suitable modifications) to run on IonQ's simulators and QPUs.